Remote Surface Classification for Robotic Platforms

Will Roderick
SUNet ID: wrtr
wrtr@stanford.edu

Connor Anderson
SUNet ID: connora
connora@stanford.edu

Aaron Manheim
SUNet ID: manheima
manheima@stanford.edu

***Abstract*—As robots move from controlled lab environments into the outside world, the ability to detect, interact with, and navigate around physical surfaces is becoming increasingly important. Previous work in texture classification using machine learning includes applying a variety of techniques to high quality image datasets in controlled settings. However, as a result of widely varying ambient conditions both indoors and outdoors, we hypothesized that transparent materials or surfaces that are not highly textured would pose a challenge to these techniques. In this project we built a device that augmented vision with additional sensing modalities, including light reflection and distance to the surface, to enable a modified bag of words algorithm to classify surface textures (such as bark, outdoor walls, indoor walls, glass, and foliage) with 98% accuracy in varying conditions with computational cost suitable for a microprocessor. In addition, we compare these results with those from training the last layer of the Inception V3 convolutional neural net with above 99% accuracy. Our results support that optimizing the first model or a simpler neural network could enable real-time classification on small robotic processors.**

I. Motivation

As robots move from controlled lab environments into the outside world, situational awareness and adaptability in uncertain and varying conditions are becoming increasingly important. In many cases, robots must be able to detect, interact with, and navigate around physical surfaces. One such application is air to surface transitions for aerial robots, which can prolong mission life for imaging, inspection, and physical data collection (see Fig. 1) [1]–[3].

Fig. 1. Aerial robot landing on an exterior building surface (from the Stanford Biomimetics & Dextrous Manipulation Lab) [4].

As for animals, robotic attachment mechanisms are specialized for specific surface types and textures, which makes surface classification an essential capability [1].

Previous work in texture classification using machine learning includes applying a variety of techniques to high quality image datasets in controlled settings [5]–[10]. For example, bi-level multi-classifiers, which use multiple standard machine learning classifiers together to make a prediction, have been used for distinguishing tiled floor, brick wall, wooden doors, and blue panels using a Canon VCC4 for robotics applications [5]. Standard texture classification methods include Local Binary Patterns, Local Phase Quantization, and Gabor Filters [6]. Some of the most popular surface and texture classification techniques include bag of visual words and convolutional neural networks (CNN's) [6]–[10]. The former typically uses image processing to extract key points from an image, cluster them using unsupervised learning techniques, and trains a support vector machine (SVM) or similar algorithm on the resulting "visual words," which represent local image features [7]–[9]. CNN's are deep learning networks with layers that act as filter banks that become more complex with depth [10].

These projects have typically used solely raw RGB photos of distinctive surfaces as inputs. However, as a result of widely varying ambient conditions both indoors and outdoors, we hypothesized that transparent materials or surfaces that are not highly textured would pose a challenge to the techniques described above. We further hypothesized that we could use surface properties to improve classification accuracy. Previous work on surface material properties has shown that materials can be classified and characterized using their reflectance [11], [12]. In addition to a camera, small robots are typically equipped with other sensors for navigation in uncontrolled environments, such as an LED and distance sensor, that can measure these kinds of properties.

In this project, we augment vision with additional sensing modalities, including light reflection and distance to the surface, to enable a learning algorithm to classify surface textures in varying conditions with computational cost suitable for a microprocessor. We focus on popular landing surfaces for aerial robots: glass, indoor walls, exterior walls, tree bark, and foliage [1]. We use two algorithms for classification: a modified bag of visual words and a convolutional neural network. The input to the first algorithm is a set of images and data on the distance to the surface. This model uses k-means clustering and SVM to make a preliminary prediction. For surfaces that are especially challenging to classify using just raw images, we use image processing techniques to extract reflectance property features and use a SVM to make more precise predictions for those surface types. We also compare the performance of this model to the more complex CNN model which takes only raw RGB frames.

II. DATASET & FEATURES

A. Data Collection System Design

We collected our dataset using a portable experimental setup that we built for this project (see Fig. 2). The device consists of a camera mounted on a tripod, LED (flash), proximity sensor, and microcomputer (Raspberry Pi). The data collection procedure consists of taking a photograph at 1080p of a surface that falls under one of our classification labels. The prototype automatically takes a picture both with and without flash in under one second. At the same time, the distance from the surface is recorded using the proximity sensor. The images and distance data are then stored on the microcomputer along with a time stamp. The wall material is recorded manually and then used for machine learning analysis.

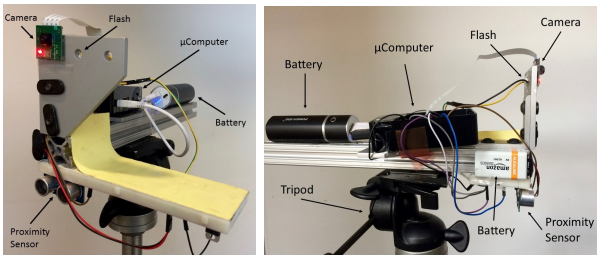

Fig. 2. Surface detection experimental setup. Left: isometric view. Right: side view.

B. Data Collection

Using this setup, we collected 1020 images of 5 surfaces: foliage, tree bark, glass, building exterior, and building interior (see Fig. 3). The photos were taken on the Stanford campus at several different times of day from morning to late afternoon in cloudy and sunny ambient conditions. The indoor photos were taken under varying lighting conditions in different buildings. The division of images into training, validation, and test sets are addressed in the *models* section.

C. Features

Both of our models use raw RGB images which are converted into feature vectors, as will be described in more detail in the *methods* section of this paper. The first model also uses image pre-processing to improve performance. In the pre-processing stage, we extract features from the subtraction of a raw RGB image from a flash enhanced image. To establish a functional baseline classification, we first focused on classifying just trees and glass to find the strength of each feature. We then expanded the feature set for indoor and outdoor surfaces.

By subtracting illuminated images from the raw image, we then can apply a normalization, conversion to greyscale, and thresholding to convert the image into a binary image (see Fig. 4 - note that not all images look this clean). Normalization is selectively performed on images containing bright pixels that may be reflections, to prevent amplifying noisy signals from otherwise static images. Thresholds are calculated based on

Fig. 3. Samples surface textures (from left to right, top to bottom: bark, indoor wall, glass, outdoor wall, and foliage).

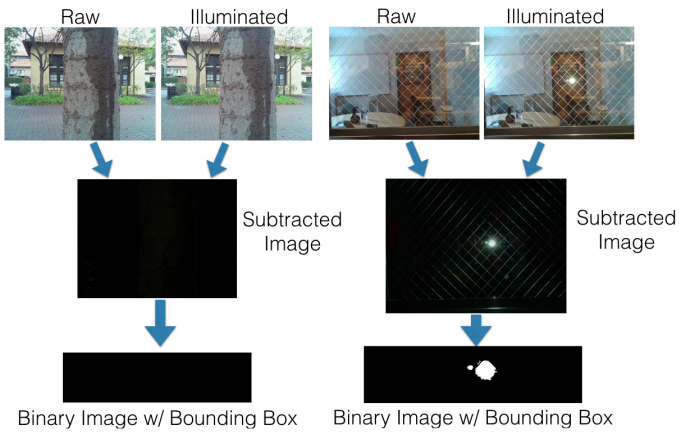

Fig. 4. Left: Image Preprocessing of Non-reflective Surface. Right: Image Preprocessing of Reflective Surface.

greyscale histograms, and selectively show only the highest intensity pixels. From the final binary image, we extract many features, and run a 10 fold cross validation analysis on a kernelized SVM which is trained on each feature. For this analysis, we perform a binary classification: reflectance or no reflectance. By removing combinations of features, one feature at a time, and recording the change in generalization error, we determine which features contribute most to the success of the SVM. From our analysis, these strongest features are maximum region area, number of distinct regions, mean pixel intensity, ratio of largest area to second largest area, and brightest pixel intensity. Fig. 8 shows an example plot of these

features on a data set of indoor and outdoor walls.

On images where a reflection is present, the subtraction step yields a bright circular region in the center of the frame. To isolate this region, a bounding box is applied to limit interference from other noise in the image (for example, a human walking behind the photo in the background). Noise from unexpected photo variations can cause significant deviations from expected results, so bounding boxes, filtering out small regions, and thresholding can all help to reduce these effects. For future versions, a high speed camera or two synced cameras would be helpful for reducing image variations due to vibrations of the camera system or robotic platform.

III. METHODS

For this project, we implemented two learning algorithms independently to compare their performance in classifying the surfaces. Each algorithm has specific advantages and disadvantages which are discussed in the following sections. The first algorithm is a modified bag of visual words and the second is a convolutional neural network. Both methods use the raw images as features. The modified bag of visual words algorithm also uses the features obtained from our image subtraction procedure.

A. Modified Bag of Visual Words (Modified BoVW)

The modified BoVW algorithm is a four step process (shown in Fig. 5). First, raw images are fed to a BoVW classification algorithm implemented in Matlab [21]. The images are broken up into a grid and between one and ten million features are extracted. These extracted features generate a 500 visual word vocabulary by utilizing k-means clustering. During this clustering step, we take the features $\{x_1, x_2, \ldots, x_m\}$ and group them into 500 clusters centered around centroids $\{\mu_1, \mu_2, \ldots, \mu_{500}\}$ according to the algorithm:

1) Initialize $\{\mu_1, \mu_2, \ldots, \mu_{500}\}$ to random values in $\mathbb{R}^n$
2) Repeat until convergence

$$\forall i \text{ set } c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

$$\forall j \text{ set } \mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

The visual words are used to train a multi-class classifier. In this classifier, the features from each training image are used in an approximate nearest neighbor algorithm with respect to their proximity to each visual word. It then generates a 500 bin histogram which counts the occurrence of each of these features in every training image. This histogram is used as the feature vector for each image.

Our BoVW model uses the error-correcting output codes (ECOC) framework in combination with binary SVMs for its multiclass predictions. This framework has been shown to greatly reduce bias and variance and thus is a good choice of multiclass classifier [13]. The model starts by creating a coding matrix of size k x n of binary classifications, where there are k possible class types and n input features. It trains multiple SVMs to populate the coding matrix with either a -1 or 1 label. Each row of this matrix is used to represent a class in binary space and is called the codeword for each class. Finally, to make a prediction, it uses the trained SVMs from the previous step to compute a codeword for the new data point (in what is called the decoding step), and finds the minimum distance between the codeword of this test sample and the codeword of each possible class [13]. Whichever class is closest is the final prediction of the model. Through this process, the classifier labels test images as either glass, indoor wall, outdoor wall, bark, or foliage.

It was observed that the BoVW algorithm has the most difficulty in distinguishing outdoor and indoor walls. In order to resolve this issue, each image that is classified as an outdoor or indoor wall is then fed through the image subtraction algorithm. This algorithm extracts features from the subtracted images including the maximum region area, the number of distinct regions, the mean pixel intensity, the ratio of largest area to second largest area, and the brightest pixel intensity. These features are correlated with the reflectance of the surface and therefore help to distinguish indoor and outdoor walls.

The extracted features along with the labels predicted by the modified bag of words are then fed to a kernelized SVM to correct the misclassifications of the BoVW. The radial basis function (RBF) is used as the kernel because it provides a good measure of the similarity between features. This is defined below where γ is inversely related to the bandwidth of the kernel:

$$K_{RBF}(x, x') = exp(-\gamma||x - x'||^2)$$

K-fold cross validation is used to determine the accuracy of the SVM (with k=10).

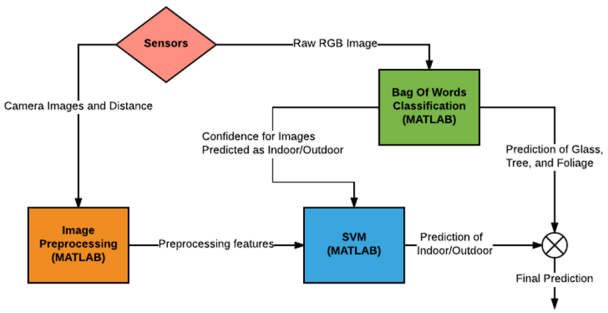

Fig. 5. Modified BoVW Flow Chart

B. Convolutional Neural Network (CNN)

Convolutional neural networks are popular machine learning tools for image classification. They are particularly useful for distinguishing variation in image texture and color [10]. In a series of layers, these algorithms apply convolutions over small windows of the image, optionally apply nonlinear (ReLU) functions ($f(x) = max(0, x)$) over the pixels of the feature maps, and pool them [14]. Training gives weights to

the connections between "neurons", or units, that make up the layers. The last layer, the output layer, gives the prediction. However, training a full state of the art neural net requires substantial computation power [15]. For example, Google's Inception V3 neural network is trained on the ImageNet ILSVRC 2012 dataset with approximately 150,000 images using 50 replicas running each on a NVidea Kepler GPU [15], [16]. We leveraged this CNN by focusing on its last layer: the softmax regression. The softmax regression is a generalized linear model that gives the predicted probability of each class being the correct output. Recent studies have shown that retraining just the last layer can be effective for classification in many applications without the added complexity of full retraining and can work well with smaller datasets [17], [18]. We used the TensorFlow library to run this neural net. This CNN is pre-trained to classify 1000 different classes of object. We trained the softmax regression layer using code found at [21]. The algorithm uses gradient descent with the associated cost function cross-entropy, defined as $H_{y'}(y) = -\sum_i y'_i log(y_i)$ where y is the predicted probability distribution and y' is the true distribution [19].

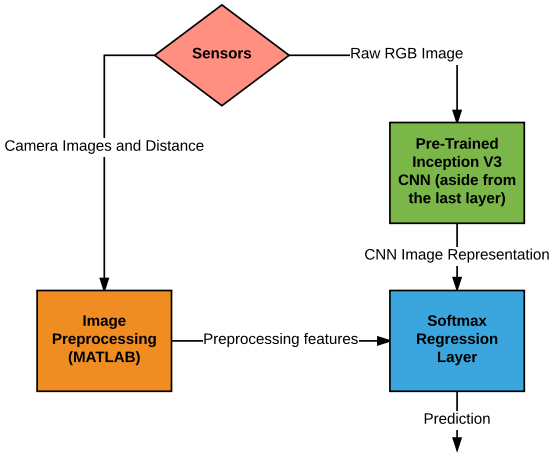

Fig. 6. Convolutional Neural Network Flow Chart

As shown in Fig. 6, the inputs to the algorithm are the RGB photos. The pre-trained layers produce a 2048 dimensional image representation vector on which the softmax layer can be trained. Our plan was to incorporate image pre-processing results into the image representation just before the last layer and retrain on our dataset, though we found that the algorithm achieved high accuracy even without the augmentation from the sensors.

IV. RESULTS & DISCUSSION

A. Modified Bag of Visual Words

The performance of the modified BoVW is based on classification accuracy. The modified bag of visual words algorithm begins with the BoVW implemented on Matlab with only the raw images as features. Hold-out validation is used to determine the accuracy of this classifier with 60% of the data set as training and 40% as testing. The BoVW is able to classify foliage, bark and glass with approximately 99% accuracy. However, indoor walls are mislabeled as outdoor walls 10% of the time and outdoor walls are mislabeled as indoor walls 3% of the time (shown in the confusion matrix in Fig. 7). To improve the overall accuracy of the modified BoVW, images predicted as indoor and outdoor walls are then fed to a kernelized SVM. Because these two classes are predominantly only confused with each other while images predicted as foliage, bark or glass are generally more accurately classified, data associated with predictions of the other classes do not go through further processing.

	Foliage	Indoor Walls	Outdoor Walls	Bark	Glass
Foliage	1.00	0.00	0.00	0.00	0.00
Indoor Walls	0.00	0.88	0.10	0.02	0.00
Outdoor Walls	0.00	0.03	0.96	0.00	0.01
Bark	0.00	0.00	0.00	0.99	0.01
Glass	0.01	0.00	0.00	0.00	0.99

Fig. 7. BovW Confusion Matrix on Raw Images

As discussed in the *methods* section, images classified as indoor or outdoor walls are pre-processed using the image subtraction algorithm. The algorithm extracts features related to the reflectance properties of the surface and feeds them through a Kernelized SVM along with the predicted classification from the BoVW.

As expected, the indoor walls reflect the flash more clearly than the outdoor walls. The reasons behind this difference are likely a result of both material properties and ambient conditions. We found that indoor walls are typically painted with a more reflective coating than the rock or rough paint of outdoor walls. In addition, outdoor ambient luminance is generally much higher than than the luminance indoors, which means that the flash was relatively more noticeable indoors than outdoors.

After experimenting with features ranging from the variance of pixel intensity to the number of regions found after processing the image (see *features* section), we found 5 features that were particularly important (shown in Fig. 8).

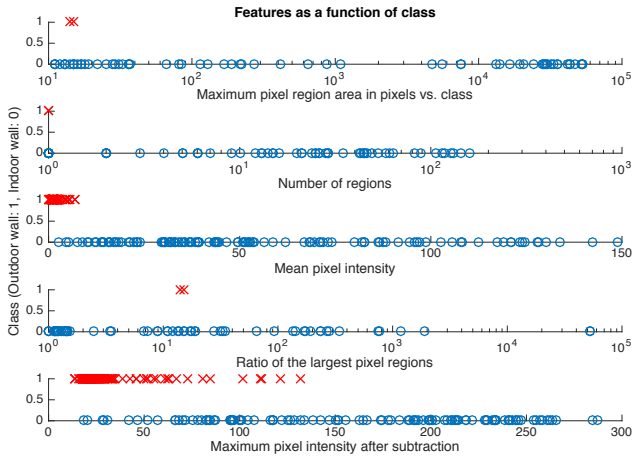

Fig. 8. Features as a function of class

We use K-fold cross validation to determine the accuracy of the binary SVM (with k=10). The SVM is able to classify indoor walls with 96% accuracy (an 8% improvement). Fig. 9 shows the updated results in a confusion matrix, which incorporates the output from the binary SVM cross-validation.

	Foliage	Indoor Walls	Outdoor Walls	Bark	Glass
Foliage	1.00	0.00	0.00	0.00	0.00
Indoor Walls	0.00	0.96	0.02	0.02	0.00
Outdoor Walls	0.00	0.00	0.99	0.00	0.01
Bark	0.00	0.00	0.00	0.99	0.01
Glass	0.01	0.00	0.00	0.00	0.99

Fig. 9. Modified BoVW Confusion Matrix

While most classes have an accuracy of approximately 99%, the lowest accuracy class is indoor walls, with 96% accuracy. Indoor walls are classified as outdoor walls 2% of the time and as bark 2% of the time. The confusion of indoor walls with non-reflective surfaces suggests that a more powerful flash and more sophisticated image processing techniques could improve that accuracy.

For this model, predictions take approximately 2 seconds to complete on an Intel i5 processor, though running in C on specialized image processing hardware could reduce this time dramatically.

B. Convolutional Neural Network

The performance of the CNN is based on classification accuracy. The learning parameters are set to their default for training the Inception V3 algorithm. The algorithm performs the training in iterative steps with a learning rate of 0.01. The training batch size per step is 100 randomly selected examples, with 100 examples for validation per step, and 500 examples for the final test prediction. The training set is kept separate from the validation and test sets.

While we anticipated the need for image processing features to boost the accuracy of the convolutional neural network, the retrained Inception V3 CNN is able to classify over 99% of the images correctly with just the raw RGB frames (see Fig 10). After only 400 training steps, we see that both the training and validation accuracy start to converge. The algorithm runs for 4000 iterations after which the predicted final test accuracy is over 99%. As we would expect, the cross-entropy drops as the algorithm converges.

To ensure that there was not an error in the data or implementation and there was no overfitting, we further tested the algorithm on cell phone images and relevant classes from the texture database from the Ponce Group at the Beckman Institute (primarily bark), which can be found at: http://www-cvr.ai.uiuc.edu/ponce_grp/data/index.html. Certain surfaces that may fall in the broad categories considered here though were not trained on, such as brick (as we focused on stucco), were not always classified correctly. Still, images of the specific surface types that we used for training were all classified correctly. This high performance may not be surprising given our dataset. The Inception V3 is pre-trained on around 1000 classes, while our dataset involves only 5

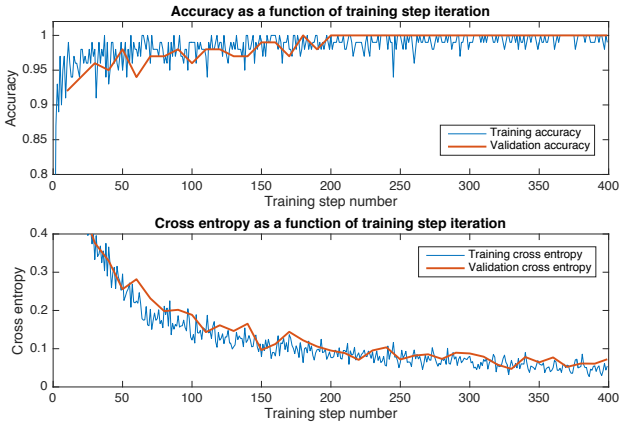

Fig. 10. (Top) Accuracy of the retrained Inception V3 neural net as a function of training step number. (Bottom) Cross-entropy, the cost function for the softmax regression layer of the neural network, as a function of training step number

classes of approximately 200 images per class in consistent and predictable conditions. An additional caveat is that real world ambient conditions may be more variable and robots may encounter surfaces that include multiple surface categories in one image. These situations were not considered here but would be valuable for future work for robotic applications. This neural network made 11 second predictions on a 2.5 GHz Intel Core i7 processor. Simpler neural networks and specialized hardware such as GPU's could reduce this time dramatically.

V. CONCLUSIONS & FUTURE WORK

Overall, image enhancement with additional sensor modalities improved the accuracy of the BoVW model to approximately 98%. Specifically, reflectance properties improved classification performance on indoor and outdoor walls. On the other hand, the neural network was able to achieve over 99% accuracy with just the raw image frames. These results indicate that while the simple model may not be as accurate as a complex neural network, the simple model is suitable for robotic surface classification. These results support that optimizing this model or a simpler neural network could enable real-time classification on small robotic processors.

Over the next weeks and months, we plan to use this system for robotic applications, including enabling aerial robots to detect viable landing locations. Specifically, we plan to optimize the simpler model for real time classification on a small processor. Farther into the future, we will focus on adaptability in cluttered environments as described in Cimpoi et al. (2015) [20] and quickly changing scenes as well as including a wider range of surfaces and ambient conditions into the classification scheme.

ACKNOWLEDGMENT

We would like to thank Professors Andrew Ng and John Duchi as well as our TA Francois Germain.

REFERENCES

[1] Roderick, William R. T., Mark R. Cutkosky, and David Lentink. "From touchdown to takeoff: at the interface of flight and surface locomotion." *Interface Focus*. 2016. In Press.

[2] Estrada, Matthew A., Elliot W. Hawkes, David L. Christensen, and Mark R. Cutkosky. "Perching and vertical climbing: Design of a multimodal robot." *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4215-4221. IEEE, 2014.

[3] Desbiens, Alexis Lussier, Alan T. Asbeck, and Mark R. Cutkosky. "Landing, perching and taking off from vertical surfaces." *The International Journal of Robotics Research* (2011): 0278364910393286.

[4] Glassman, Elena, Alexis Lussier Desbiens, Mark Tobenkin, Mark Cutkosky, and Russ Tedrake. "Region of attraction estimation for a perching aircraft: A Lyapunov method exploiting barrier certificates." In *Robotics and Automation (ICRA), 2012 IEEE International Conference* on, pp. 2235-2242. IEEE, 2012.

[5] Martnez-Otzeta, Jos Mara, Basilio Sierra, and Elena Lazkano. "Image analysis and automatic surface identification by a bi-level multiclassifier." *In International Symposium on Brain, Vision, and Artificial Intelligence*, pp. 467-476. Springer Berlin Heidelberg, 2005.

[6] Hafemann, Luiz G., Luiz S. Oliveira, and Paulo Rodrigo Cavalin. "Forest Species Recognition Using Deep Convolutional Neural Networks." *In ICPR*, pp. 1103-1107. 2014.

[7] Zhang, Jianguo, Marcin Marszaek, Svetlana Lazebnik, and Cordelia Schmid. "Local features and kernels for classification of texture and object categories: A comprehensive study." *International journal of computer vision* 73, no. 2 (2007): 213-238.

[8] O'Hara, Stephen, and Bruce A. Draper. "Introduction to the bag of features paradigm for image classification and retrieval." *arXiv* preprint arXiv:1101.3354 (2011).

[9] Yang, Jun, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. "Evaluating bag-of-visual-words representations in scene classification." In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pp. 197-206. ACM, 2007.

[10] Andrearczyk, Vincent, and Paul F. Whelan. "Using Filter Banks in Convolutional Neural Networks for Texture Classification." *arXiv* preprint arXiv:1601.02919 (2016).

[11] Bennett, H. E. J., and J. O. Porteus. "Relation between surface roughness and specular reflectance at normal incidence." *JOSA* 51, no. 2 (1961): 123-129.

[12] Dana, Kristin J., Bram Van Ginneken, Shree K. Nayar, and Jan J. Koenderink. "Reflectance and texture of real-world surfaces." *ACM Transactions on Graphics (TOG)* 18, no. 1 (1999): 1-34.

[13] Kong, Eun Bae, and Thomas G. Dietterich. "Error-Correcting Output Coding Corrects Bias and Variance." *ICML*. 1995.

[14] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." In European Conference on Computer Vision, pp. 818-833. Springer International Publishing, 2014.

[15] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." *arXiv* preprint arXiv:1512.00567 (2015).

[16] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

[17] Hafemann, Luiz G., Luiz S. Oliveira, Paulo R. Cavalin, and Robert Sabourin. "Transfer Learning between Texture Classification Tasks Using Convolutional Neural Networks." *2015 International Joint Conference on Neural Networks (IJCNN)* (2015): 1-7.

[18] Donahue, Jeff, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition." *In ICML*, pp. 647-655. 2014.

[19] Google Brain Team. "MNIST For ML Beginners — TensorFlow." *TensorFlow*. Google, 13 Dec. 2016. Web. https://www.tensorflow.org/tutorials/mnist/beginners/ (accessed 16 Dec. 2016).

[20] Cimpoi, Mircea, Subhransu Maji, and Andrea Vedaldi. "Deep filter banks for texture recognition and segmentation." *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3828-3836. 2015.

[21] Code/libraries used: 1. Matlab BoVW image classifier: https://www.mathworks.com/help/vision/ref/trainimagecategoryclassifier.html 2. Tensorflow Inception retraining: https://github.com/tensorflow/tensorflow